The DSN Programming System

G. A. Madrid
TDA Engineering Office

This article describes the DSN Programming System and its current status. Accomplishments and plans are summarized. A study to revise the system design is in progress.

I. Introduction

The DSN Programming System is an interrelated body of procedures, tools, equipment, and information designed to assist DSN software developers and managers in the efficient and effective production of software for DSN tasks. The major goals of this system are (1) to reduce the cost of producing, documenting, and maintaining DSN software, and (2) to improve the ability of managers to cost, schedule, monitor, and evaluate DSN software-development tasks under their purview.

The DSN Programming System is of particular importance to the Mark IV-A Implementation Project due to its need to maintain strict budgetary and schedule constraints on all development being performed under their aegis. It is expected that the Programming System will provide just this type of assistance in the development of computer software for the new series of computers replacing those currently at the DSN deep space stations (MODCOMP II computers are being replaced by MODCOMP CLASSICs).

II. System Objectives

To achieve the stated goals, the DSN Programming System has set for itself the following objectives:

(1) To provide implementing organizations the standard practices and software-development methodology (Refs. 1 and 2) necessary to increase the cost effectiveness of the software-development process. The increase will be reflected in reduced overall life-cycle costs due mainly to improvements in maintainability and reliability of software developed using common standards.

(2) To develop and provide the standard languages and software-development tools needed to code, check out, and document computer programs for the DSN. These improvements in the developmental environment will result in improved productivity and personnel morale. These are major ingredients in developing software on schedule and within budget constraints.

(3) To develop and provide computer-generated aids and reports needed to increase management visibility into the software-development process. This increased visibility will permit a more efficient use of available resources, a timely recognition of problems, and the information required to reduce the risk in software-management decisions.

(4) To assure, by means of an appropriate functional design, accord of the Programming System with the actual software-development process. This will permit an evaluation of the system based on its functional flow and more clearly delineate the interfaces and functions required by the system to support the software-development process.

(5) To develop the methodology and procedures needed to measure system performance against realistic performance criteria. This methodology needs to be a permanent fixture in any new system design if that system is expected to remain a viable entity for any length of time.

III. System Description

The elements and structure of the current DSN Programming System are shown in Fig. 1. It is, in design and approach, very much the same system described by Hodgson (Ref. 3) and reported by Irvine (Ref. 4). The basic structure is defined in five basic categories:

(1) Software Standard Practices.

(2) Standard Processors.

(3) Programming System Data Base.

(4) Management Aids.

(5) Implementation Aids.

This structure, however, does not provide a functional design for the system and therein lies its major deficiency. Section V discusses this problem and the study underway to redefine the system.

The Software Standard Practices segment of the system is represented by a series of documents that specify standard methodology, guidelines, and practices dealing with the implementation of software from inception to operations. These documents are maintained in the 810-XX series of DSN documentation. However, a compendium of these standards (Ref. 5) also exists and may be more suitable for those readers merely desiring to inform themselves regarding the DSN Standard Practices. The major activity in this area currently involves reviewing these standards in light of their effectiveness and usefulness to the implementing organizations. Modifications are made as required contingent on appropriate review board approval.

The Standard Processor segment of the system is responsible for developing and maintaining the standardized software environment needed by the software developers and managers of software-implementation efforts. This function was responsible for selecting and implementing the two current Standard Languages: MBASIC* and HAL/S. The former is termed the "off-line" language to be used for administrative and nonreal time applications. The latter is termed the "on-line" language to be used for real-time DSN applications. Procurement of other useful tools and languages are also performed under the purview of this system element.

The Implementation Aids segment of the system is an adjunct to the Standard Processor segment. Its main purpose is to provide the software developers additional tools to assist them in designing, documenting, and testing computer programs. The Data Base and Management Aids segments are also complementary in that their purpose is to provide the management aids, reports, and information required to improve implementation process visibility and ensure informed management decisions at critical points in the development effort.

IV. Status and Plans

This section discusses the status and plans of those segments of the system where activity or changes occurred since the last report (Ref. 3).

A. Software Standard Practices

Standards for the production of System and Subsystem Functional Requirements are in draft form. Finalization of these documents is being placed in abeyance pending a formal review of the Programming System Functional requirements and design. Such a review is expected in the first half of calendar year 1982 (see Section V).

B. Standard Processors

The majority of activity in the Programming System has dealt with the implementation of the "on-line" language, HAL/S. This language has been identified by the DSN as the standard real-time language to be used in any software development for the real-time computers being installed at its remote stations under the Mark IV-A Implementation Plan.

A version of the HAL/S Compiler for listing on a MODCOMP 7870 computer was developed by Intermetrics, Inc., under contract to JPL. A completed version of this compiler was accepted by JPL on May 14, 1981. Delivery was accepted despite some errors found during testing of the product. Of the original errors found, only four were considered major problems, and all of these have been resolved by Intermetrics under a maintenance contract with JPL. There are currently 22 outstanding problems of which 2 are considered of major importance. All of these are due to be resolved by January 1982.

A MODCOMP 7870 with operating system and peripherals was purchased and installed in October 1980 to serve as the host development computer for the HAL/S language and its associated tools.

*A trademark of the California Institute of Technology.

Training on use of the new compiler language has begun. A training class conducted by Intermetrics was held on July 20, 1981, for 25 programmers and supervisors who will be utilizing the language in their implementation tasks. Additional classes are planned.

Development of a more comprehensive set of tests for the HAL/S compiler is underway. Results from these tests will be used as a condition for the transfer of the compiler to operational status. Such a transfer is not expected until the first quarter of calendar year 1982 when it is expected that the number of problems uncovered through testing will have stabilized. This date will not impact Mark IV-A software development since a code can be developed using the current compiler and then recompiled with the operational version when that is made available.

A list of HAL/S-compatible tools and aids available from other NASA centers has been compiled for use in determining how to improve the DSN development environment as quickly and as cost effectively as possible. Active participation of TDA and implementing organization personnel in the NASA HAL/S User and Configuration Control Board is being encouraged so that repetition of costly mistakes can be avoided through the sharing of information with HAL/S users at other NASA centers. This could lead to reduced development costs and reduce the impact of unnecessary language changes on the users.

The DSN Standard "off-line" language, used for nonreal time purposes, is being maintained for current users on the JPL institutional computers (Univac 1100/81). A study to replace this standard with ANSI-BASIC is being proposed. The decision for the computer host of this language should be subsidiary to the more fundamental decision of the functional design of the Programming System itself.

C. Management Aids

A preliminary version of a Work Breakdown Structure (WBS) report generation system has been made available to management. This management aid has elements on both the AODC (Automated Office Data Center, a micro-processor computer system developed by JPL) and the MODCOMP 7870 currently being used for the HAL/S development. A Software Cost Estimation Model (SOFTCOST) is similarly hosted on the AODC with data access through the interconnection with the MODCOMP 7870.

Current planning is directed toward providing the management functions with a separate MODCOMP 7870 thus freeing the development computer for more on-line use by software developers. The management computer would then be the central node for the network of terminals and processors currently used by management to maintain data bases, estimate cost, report productivity, and produce up-to-date schedules and resource allocations.

V. Future Plans

The use of the Programming System to support the Mark IVA Implementation effort has indicated a need to review the design and structure of the current system.

The lower-level functional elements of the system are well defined and can be directly related to requirements or constraints. These lower-level elements, however, are not organized along functional lines but along generic lines. If the system is to support the Mark IV-A development effort, a better approach would be to organize these elements in conformance with the actual functional flow of the development process. This will permit a more conventional division of the system into subsystems as well as a clearer definition of the interfaces between subsystems and the interaction between elements. Such a redesign would permit a clearer evaluation of priorities and the impact of changes on the entire system would be more clearly discerned.

A study to propose a revised system design along these lines is underway and will be reviewed and presented in the first half of calendar year 1982. It is also planned to provide a long-term plan that will guide the system development for the next five years and provide some insight into future development through 1992.

References

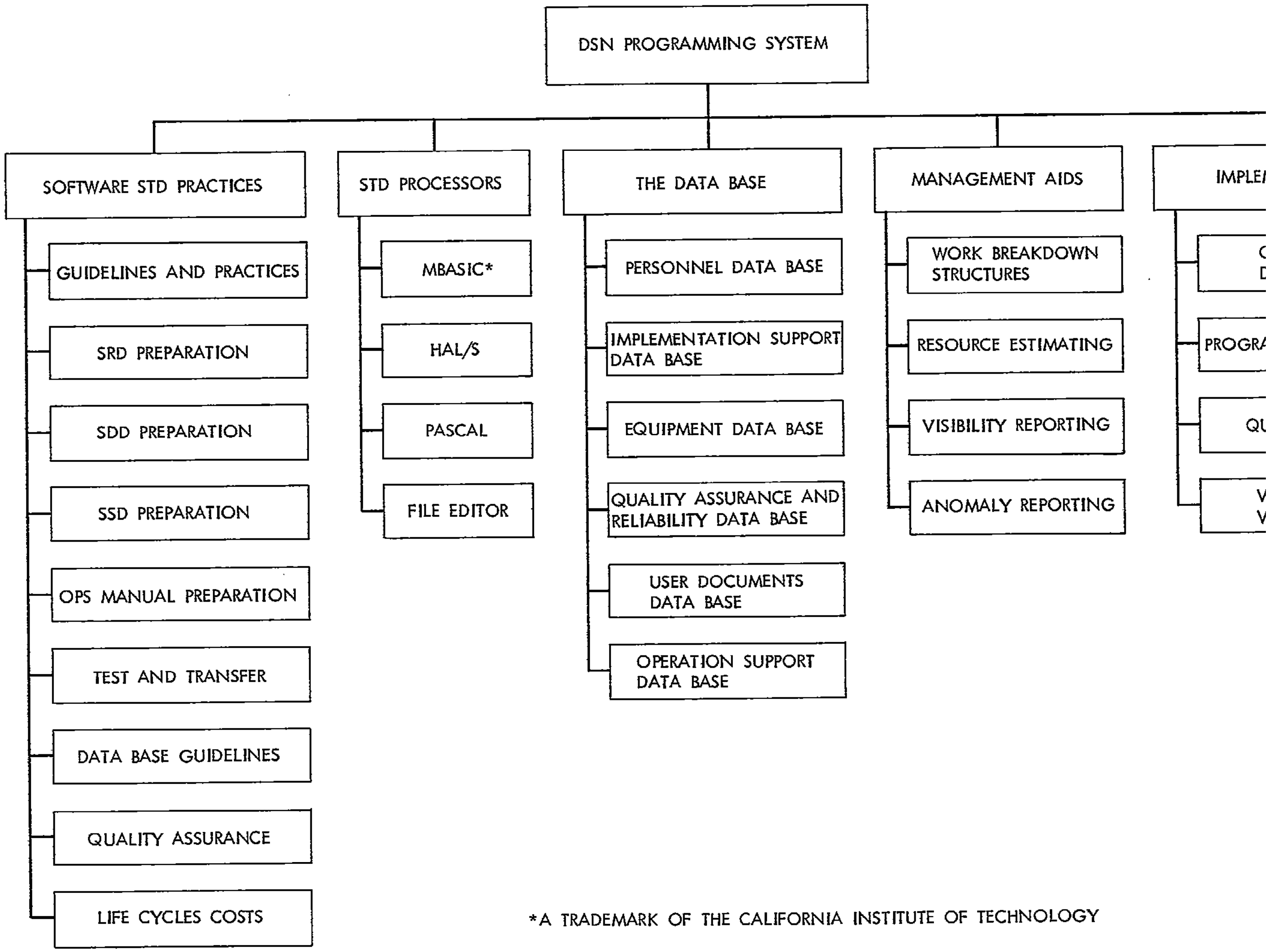
1. Tausworth, R. C., *Standardized Development of Computer Software, Volume II: Standards*. Special Publication 43-29, Jet Propulsion Laboratory, Pasadena, Calif., Aug. 1977.

2. Irvine, A. P., and McKenzie, M., "Evaluation of the DSN Software Methodology," in *The Deep Space Network Progress Report 42-48*, pp. 72-81, Jet Propulsion Laboratory, Pasadena, Calif., Dec. 15, 1978.

3. Hodgson, W. D., "The DSN Programming System," in *The Deep Space Network Progress Report 42-41*, pp. 4-9, Jet Propulsion Laboratory, Pasadena, Calif., Oct. 15, 1977.

4. Irvine, A. P., "The DSN Programming System," in *The Deep Space Network Progress Report 42-50*, pp. 4-6, Jet Propulsion Laboratory, Pasadena, Calif., Jan. – Feb. 1979.

5. Irvine, A. P., ed., *Standard Practices for the Implementation of Computer Software*, Publication 78-53, Jet Propulsion Laboratory, Pasadena, Calif., Sept. 1978.



*A TRADEMARK OF THE CALIFORNIA INSTITUTE OF TECHNOLOGY

Fig. 1. Structure and elements of the DSN Programming System